Applied Soft Computing Strategies for Autonomous Field Robotics

Edward Tunstel, Ayanna Howard, Terry Huntsberger, Ashitey Trebi-Ollennu and John M. Dolan*

NASA Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91109, USA

**The Robotics Institute*
Carnegie Mellon University
Pittsburgh, PA 15213, USA

Appears in: C. Zhou, D. Maravall, and D. Ruan (Eds.), ***Autonomous Robotic Systems: Soft computing and hard computing methodologies and applications***, Studies in Fuzziness and Soft Computing Series, Physica-Verlag, Heidelberg; New York, 2003, pp. 75–102.

Applied Soft Computing Strategies for Autonomous Field Robotics

Edward Tunstel[1], Ayanna Howard[1], Terry Huntsberger[1], Ashitey Trebi-Ollennu[1], and John M. Dolan[2]

[1] NASA Jet Propulsion Laboratory, Caltech, Pasadena, CA 91109, USA
[2] Carnegie Mellon University, Pittsburgh, PA 15213, USA
tunstel@robotics.jpl.nasa.gov

Abstract. This chapter addresses computing strategies designed to enable field mobile robots to execute tasks requiring effective autonomous traversal of natural outdoor terrain. The primary focus is on computer vision-based perception and autonomous control. Hard computing methods are combined with applied soft computing strategies in the context of three case studies associated with real-world robotics tasks including planetary surface exploration and land survey or reconnaissance. Each case study covers strategies implemented on wheeled robot research prototypes designed for field operations.

1 Introduction

Hard computing methods which address robot perception and control issues rely upon strong mathematical modeling and analysis [1,2]. The various approaches proposed to date are suitable for control of industrial robots and automatic guided vehicles that operate in structured environments and perform relatively well-defined repetitive tasks, such as manipulator positioning or tracking fixed/pre-programmed trajectories. Operations in unstructured environments, on the other hand, require robots to perform more complex tasks for which sufficient analytical models for control are often difficult to develop. This is typically the case in field robotics, which is concerned with development and application of robotic systems as tools for performing tasks or missions in unstructured, demanding, and/or hazardous natural environments (e.g., land surfaces and subsurfaces, sea, air, space, etc.). For applications in which analytical models are available, it is questionable whether or not the models are complete, or whether uncertainty and imprecision are sufficiently accounted for [3,4]. These issues become very significant in applications of autonomous field and space robotics [5], where target environments are not always known in sufficient detail to enable robust robotic system performance using hard computing techniques alone.

Field robotic vehicles intended for operation on land surfaces are the focus of this chapter. The target environments consist of natural rugged terrain, as opposed to relatively flat ground surfaces such as paved roadways. For the latter case (as with indoor environments), the motion controls for robot mobility systems can often be designed based on linear system models, or simpified nonlinear models of vehicle kinematics or dynamics assuming motion on a 2-D plane [2]. Practical mobility control systems for use in outdoor rugged terrain must account for a wider array of real-world complexities [6], the most fundamental of which is the fact that complex motions in the third dimension occur quite frequently. The situation is similar for rovers that interact with complex planet surfaces by roving across or burrowing through terrain [7,8]. For the robotic spacecraft that delivers them to the surface of remote planets, conventional estimation and control techniques have proven quite useful [9]. This is due to the fact that the physical laws of orbital mechanics and planetary atmospheric aerodynamics are reasonably well understood and well-behaved in space. Unfortunately, the complexities of terrestrial surface mobility and navigation recur once rovers are deployed on remote planet surfaces (and are sometimes compounded by reduced-gravity effects).

Autonomous mobile field robots must be designed to handle complex, and often uncharted, terrain encountered through the course of navigation, while remotely situated relative to human supervisors. Robust mission execution requires systems that can autonomously, and with minimal supervisory communication from humans, operate in natural environments and perform goal-directed tasks. Robotics researchers at the NASA Jet Propulsion Laboratory (JPL) have integrated soft computing techniques with more conventional hard computing methods to address some of the problems related to surface exploration of planet Mars using autonomous rovers, as well as control problems specific to terrestrial land vehicles. Soft computing strategies for rover perception, navigation decision making, and control have been developed and applied to complement conventional methods in an effort to achieve required levels of robustness. These include applications of fuzzy logic and neural network computing methods that facilitate reasoning and action selection in unfamiliar environments in order to enable reliable and safe mission execution. Using three case studies, several approaches are presented in this regard that have been integrated on field robot research prototypes and validated through operation on natural terrain. Each case study is presented following a brief overview in the context of general field mobile robot problems.

2 Case Study Overview

Autonomous mobile robot computational systems include functional and/or behavioral components that process sensory data, and maintain internal state information in order to perform repetitive cycles of processing designed to achieve specified tasks. Repetitive activities typically include the following: generation of perceptions (and sometimes models) of the target environment, utilization of sensing and perceptions to compute intelligent navigation or motion planning decisions, and execution of motion decisions via automatic control of actuators. This is the so-called *sense-plan-act* cycle common to many robotics tasks. Three isolated case studies are presented below as representative examples of how soft computing strategies have been applied to address some important problems, associated with this cycle, that arise in field robotics applications. Each covers a specific field robot implementation of the strategies applied. The implementation target in each case is a different wheeled mobile robot, and although the cases are isolated, they share the following common thread. Each case study addresses a function typically included in the repetitive cycle of activities associated with the overall task of effectively traversing natural terrain for purposes such as planetary surface exploration, land survey, and reconnaissance. The techniques are also relevant to other real-world field applications involving military and search/rescue operations, as well as agricultural, construction, and mining automation. The overall focus here is on visual perception and intelligent control. In particular, the case studies cover: (1) stereo vision-based perception for autonomous navigation, (2) monocular vision-based perception for effective mobility, and (3) control synthesis for locomotion systems complicated by nonlinearity and/or lack of suitable mathematical models. A thorough coverage of complementary soft computing-based navigation techniques can be found in [3].

Case studies 1, 2, and 3 center around a prototype planetary rover, a commercially available mobile robot research platform, and an unmanned ground vehicle, respectively. The intended application environment for case studies 1 and 2 is a barren landscape with soils of various consistencies and cluttered with rocks, such as the surface of planet Mars. For case study 3, the application environment may consist of unpaved roads, foliage, and/or grass. In all cases, the application environments are neither engineered nor structured in any way to accommodate the field robots. To overcome some of the practical problem constraints associated with task complexity and environmental challenges, it is sometimes advantageous to augment conventional hard computing methods with soft computing techniques. The case studies detail three instances for which this was achieved in the respective computing implementations for robot perception and control. Prior to field deployment, applied soft computing strategies must undergo extensive testing to ensure that the host robotic systems function properly, and are reliable enough to survive mission duration and achieve mission success. Advanced technologies are therefore developed in stages leading to increased readiness for field application; special concerns are measured against specific requirements of particular tasks/missions. Case studies presented herein reflect the present state of development of several soft computing strategies, and the scope of each is limited to overviews of ongoing technical research. Sections 3 and 4 present the case studies covering practical approaches to intelligent vision-based perception of natural terrain for autonomous navigation and mobility. This is followed by coverage of an adaptive speed control solution for field locomotion in Section 5. Each section describes the problem(s) addressed and identifies the hard computing and soft computing aspects of the applied solution. Motivation for adopting particular soft computing techniques is discussed, followed by descriptions of the implementation.

3 Case Study 1: Stereo Processing for Navigation

In 1997, the successful NASA Mars Pathfinder mission demonstrated that geological science exploration tasks can be performed within short range of a lander using small lightweight rovers with modest computational resources [10]. On future missions, more complete coverage of the planetary surface will require the development of higher level autonomy onboard the rovers. However, it is likely that advanced autonomy capabilities will also have to be realized using modest computational resources. This is due to practical limitations on mass and power typical of space flight avionics, as well as limited processing capabilities characteristic of radiation-hardened CPUs. A key component of required advanced autonomy is long-range navigation with obstacle detection and avoidance capabilities. Traditional methods used for achieving such autonomy often rely on range maps generated from a stereo image pair. Depending on the implementation, stereo image processing can consume significant portions of the time required to complete the sense-plan-act cycle, and/or require relatively large computational resources. For long-range rover missions, fast and efficient algorithms are desirable for reducing the amount of time required for stereo processing. A prototype technology rover called Sample Return Rover (SRR) is used for development and testing of such algorithms at JPL (Fig. 1). The sensor suite for navigation consists of a stereo pair of hazard cameras body-mounted to the front of the rover, and a single goal and stereo pair of cameras mounted to an articulated mast. This case study addresses algorithm designs for the generation of fast obstacle detection and avoidance using the front hazard cameras.

A variety of computational methods apply to stereo processing including maximum likelihood [11], dynamic programming [12], and biologically-inspired methods based on neural networks [13] and wavelets [14]. This case study employs a fuzzy self organizing feature map (FSOFM) algorithm combined with wavelet image processing to generate navigation commands for a behavior-based control system called BISMARC (Biologically Inspired System for Map-based Autonomous Rover Control). The system architecture for BISMARC is shown in Fig. 2. The stereo processing system does not build range maps as is done traditionally, but instead relies on a raw encoding of the image disparity information for reactive action generation.

3.1 Motivation and Approach

Striate cortical cells in cats have been found to respond to both monocular and binocular inputs [15]. In addition, cells in the retinal mammalian visual system respond to small lines and edges [16]. The behavior of these cells can

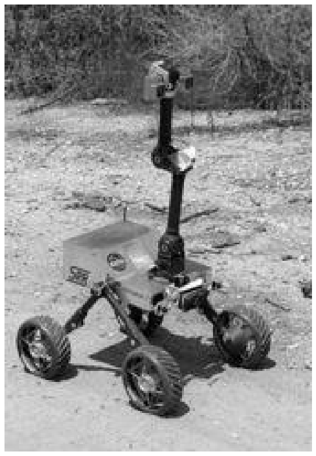


Fig. 1. SRR, a JPL rover.

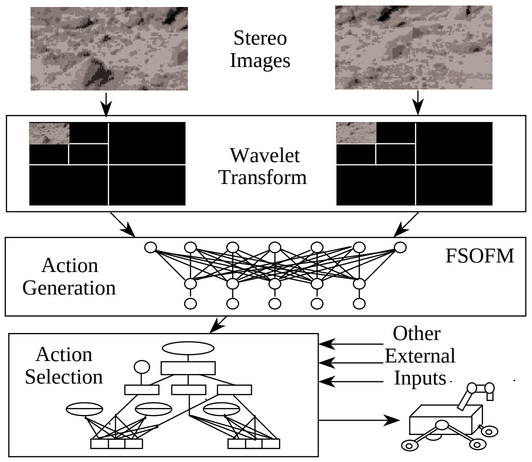


Fig. 2. BISMARC system architecture.

be modeled using Gabor wavelets by determining the maximal response of filters that are tuned to specific frequency and orientation [14]. This method of analysis detects salient features that correspond to points of significant curvature change in the image. The vision preprocessing level for BISMARC uses such wavelet-based algorithms to decompose the images generated by each of the stereo cameras. The wavelet decomposition provides information about the scale, location, and orientation of features in an image. Because the wavelet decomposition contains information about the local frequency content of an image, it can represent visually important features (such as edges) more compactly than many of the other transforms commonly used in image processing [17].

After the wavelet transform is performed on each of the stereo images, a vector is formed using the multiresolution information from the two highest levels of the wavelet horizontal and vertical detail channels. This vector of length $\frac{5n^2}{4}$ elements, where n is the image width, is the input to the FSOFM — the output being any of six action states: go forward, backup, turn right, turn left, stop at goal, or pick either direction to turn. Raw stereo visual information is being encoded without any attempt to label individual features or objects beyond the desired action associated with the input pattern. Although the FSOFM is an unsupervised network, it can be trained by presenting it with samples and labeling the output nodes that correspond to each of the generated actions. The weights are then clamped for all subsequent runs.

The process of depth analysis can be functionally simulated using a combination of wavelet-based preprocessing coupled with a neural network that responds to binocular inputs. However, computation of Gabor wavelet coefficients for the full range of frequencies and orientations necessary to completely encode an image is computationally expensive, which limits their usefulness for fast image processing. There is another class of wavelet transforms that is more suited for constrained rover computing environments.

3.2 Wavelet and Neural Network Implementation Strategy

The standard wavelet decomposition of a general signal (or image) f is a representation in terms of linear combinations of translated dilates of a single function ψ:

$$f(x) = \sum_{\nu} \sum_{k} c_{\nu,k} \psi_{\nu,k}(x), \tag{1}$$

with $\psi_{\nu,k}(x) = 2^{\nu/2}\psi(2^{\nu}x - k)$ for integers ν and k. The mapping that takes f into the corresponding coefficients $c_{\nu,k}$ is called the wavelet transform of f. Since the transform is linear, all of the information content of the image or signal is contained in the $c_{\nu,k}$s. The detail channels of the coefficient space are defined as the orientation-specific difference between average channels at resolution ν and $\nu - 1$. A hierarchical representation of f is obtained by varying the scale from coarse (small ν) to fine (large ν).

There are a number of limitations to standard wavelets, including border artifacts and a dyadic (power of 2) restriction on image sizes. A method that can be used to avoid these problems involves the use of biorthogonal wavelets [18,19]. A biorthogonal wavelet decomposition uses two families of basic building blocks $\psi^{i}_{\nu,k}$ and $\tilde{\psi}^{i}_{\nu,k}$ to decompose the image as

$$f(x) = \sum_{i finite} \sum_{\nu} \sum_{k} \left\langle f, \tilde{\psi}^{i}_{\nu,k} \right\rangle \psi^{i}_{\nu,k}, \tag{2}$$

where $\left\langle f, \tilde{\psi}^{i}_{\nu,k} \right\rangle$ is defined as the inner product of f and $\tilde{\psi}^{i}_{\nu,k}$. This type of wavelet can be defined to exist within an arbitrarily shaped region [18,19]. We have previously used this form of the transform for texture analysis in adaptive, stereo range map generation for local rover navigation [20]. An example of the biorthogonal wavelet transform applied to a stereo pair of images from SRR's hazard cameras is shown in Fig. 3, where the hierarchical display format of Mallat is used [17]. The set of coefficients that correspond to the average channel at the coarsest resolution ν is in the upper left hand corner of the figure. The coefficients corresponding to the vertical, diagonal, and horizontal detail channels are shown in clockwise order from the upper right hand corner of the figure, and at different levels of resolution while traveling from the lower right to the upper left corner.

Fuzzy Self-Organizing Feature Map The neural network model [21] used for the action generation process is a modification of the self-organizing feature maps of Kohonen [22]. It is also a member of the generalized class of clustering networks developed by Pal, Bezdek, and Tsao [23] (hereafter referred to as GLVQ). The FSOFM, shown in Fig. 4, consists of three layers.

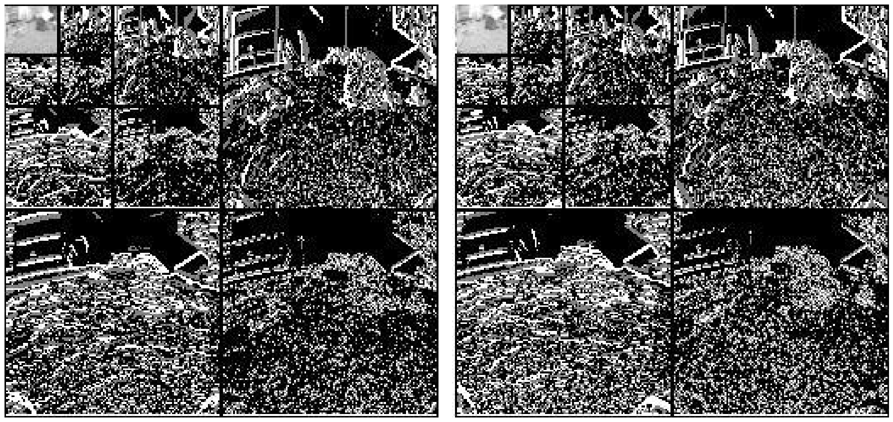


Fig. 3. Wavelet processed stereo pair used as FSOFM preprocessing step.

The input layer feeds forward into a distance layer which determines the distance between the input vector and the current weights using a predefined metric. The distance layer then feeds forward into a membership layer which calculates the membership values $u_{ji} \in [0.0, 1.0]$ of the input vector to the set of all output vectors. This is done using the following form derived from the fuzzy c-means algorithm [24] and used in a previously developed computer vision system [25]:

$$u_{ji} = \frac{1}{\left[\sum_{l=0}^{j-1} \left(\frac{d_{ji}}{d_{li}}\right)^{\frac{2}{(m-1)}}\right]} \tag{3}$$

where j is the number of output nodes, d_{ji} is the distance between the input vector ξ_i and weight vector W_j, and $m \in [2.0, \infty)$ is a weighting exponent. These membership values u_{ji} are then fed back into the network and participate in the weight update rule as:

$$W_j = W_j + u_{ji} * dw_j \tag{4}$$

where $dw_j = (\xi_i - W_j)$. The addition of a feedback loop allows the network to respond to the localized patterns of activity in the distance layer. The sum of membership values for any given input vector to all of the j sets is normalized to 1.0. This means that an input vector that is not close to any of the previously defined sets (an outlier) will have an equal membership value to all sets approximately equal to $1/j$. Such a vector would be classified as unfamiliar. The complete algorithm is given as [26]:

P1. Randomly initialize weights $W_j \ \forall \ j$ to values between 0 and 1. Set the *limit* value to be sufficiently small, e.g., 0.001, and set *total_diff* to 0.

P2. For each vector input ξ_i, $i = 1, 2, ..., n$ where n is the number of inputs:

   1. Calculate d_{ji} and determine the feedback membership value u_{ji} for each input vector using Eq. (3).

2. Update the weight change factor dw_j such that $dw_j = (\xi_i - W_j)$.
3. Save the current weight W_j as W_old_j before updating weight.
4. Update weight W_j using Eq. (4).
5. Return the value $diff$, where $diff = \sum(W_j - W\_old_j)^2$. Update $total\_diff$ using $total\_diff = total\_diff + diff$.

P3. If $total\_diff > limit$ then go to P4, else go to P5.
P4. Reset $total\_diff$ to 0 and go to P2.
P5. Write out weight file and determine the fuzzy membership value u_{ji} for each input vector ξ_i by using Eq. (3).

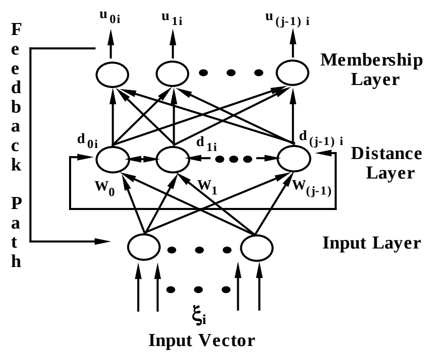


Fig. 4. Fuzzy self-organizing feature map neural network.

The FSOFM algorithm has certain advantages over GLVQ [23] in that there is no need to determine a winning output node. This property of FSOFM leads to a uniform weight update rule for all nodes. Details and performance of sequential and parallel versions of the algorithm can be found in the original paper [26]. Experimental studies indicated that the network reproduced the segmentation results from an earlier standard implementation of the fuzzy c-means algorithm [25] with close to two orders of magnitude increase in performance.

The network is trained with a set of wavelet processed images that are a representative sample of the types of obstacles that a rover would encounter. The right image from some of the stereo pairs (taken by hazard cameras on SRR) that were used to train the navigation FSOFM are shown in Fig. 5. The actions generated by the FSOFM for each stereo image pair are clockwise from top left: turn right, turn left, backup, go either way, stop at goal, and go forward. A total of five hundred stereo pairs were used for the training session, which took 637 epochs to converge. This should be compared to a back-propagation implementation, which typically takes hundreds of thousands of epochs to converge. Recall of the trained images was 100%. An advantage of using the FSOFM for the action generation level lies in the membership values that are generated at the output nodes. The sum of these values is normalized to one, and the relative size of the membership values gives a ranking

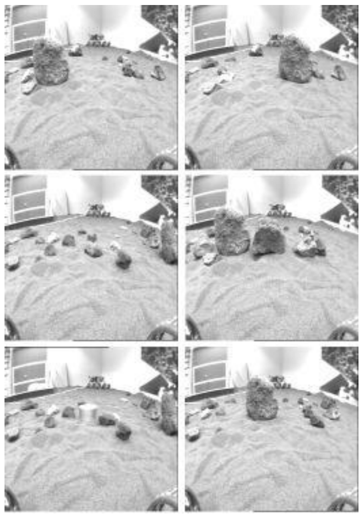


Fig. 5. Right stereo images from FSOFM training samples.

of the actions that are possible. The response to a unknown ambiguous input would be 0.16 from every output node, which means that any output greater than this is the favored action, with the largest membership value being favored. The response of the network to inputs that it was not trained with are shown in Fig. 6. Actions (membership values) that were generated for the images from left to right in the figure are: turn left (0.52), go forward (0.41), and turn right (0.37). The middle image also returned a membership value of 0.27 for turn left, which indicates that movement towards the left will be necessary to avoid the obstacle in the long run. Movement far from an obstacle is automatically generated, and the rover will only turn when its field of view has the vector of wavelet coefficients that indicate looming of an obstacle large enough to require avoidance behavior.

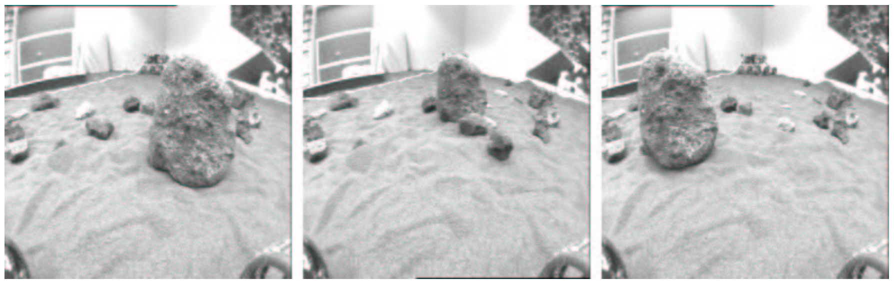


Fig. 6. Actions returned from FSOFM with unknown inputs. Although the system was not trained with these images (right image from stereo pair shown), it was able to generalize to the appropriate actions.

In this application, the FSOFM is employed to provide useful information for collision-free navigation. In particular, the FSOFM returns membership values that give a strong indication of the obstacle content of the local environment. To date, the wavelet and neural network strategies have been used to navigate the SRR platform in laboratory testbeds comprised of sand and rocks. Additional phases of testing and validation in uncontrolled outdoor terrain are planned for the near future. This case study focuses on a high-level vision-based strategy for terrain perception using efficient stereo image processing to facilitate rover navigation. In the following section, Case Study 2 describes a terrain perception strategy that employs low-level monocular vision for maintaining mobility performance during navigation via supervisory-level control of vehicle speed and terrain-aided position estimation on varied terrain surfaces.

4 Case Study 2: Terrain-Aided Mobility

To achieve effective mobility in natural terrain, it is desirable for motion controllers to accommodate physical interactions between the mobility system and rugged terrain while maintaining vehicle safety and reasonably accurate position estimation. While a rover traverses outdoor terrain, perception of the type and condition of the terrain surface provides clues for safe mobility assessment. Human automobile drivers are able to perceive certain road conditions (e.g., oil slicks, pot-holes, and ice patches) as measures of safety, and react to them in order to reduce the risk of potential accidents. In a similar manner, rover potential safety can be inferred and reacted on the basis of knowledge about the terrain type or surface condition. Two prevalent effects of wheel-soil interactions are wheel slippage on low tractive surfaces and wheel sinkage on soft surfaces. On dry paved roads, traction performance is maximal for most wheeled vehicles due to the high coefficient of friction/adhesion between the road and the tread. On off-road terrain, however, a variety of surface types are encountered on which rover wheels are susceptible to slippage. Excessive wheel slippage reduces the effective traction that a rover can achieve and, therefore, its ability to make significant forward progress. On soft soils, such as fine-grained sand, excessive wheel slippage can often lead to wheel sinkage and eventual entrapment of the vehicle. Frequent loss of traction during a traverse from one place to another will also detract significantly from the ability to maintain good position estimates for rover localization. Non-systematic localization errors due to wheel slip are compounded by errors due to wheel sinkage. As the load-bearing strength of the terrain/soil varies so does the amount of wheel sinkage. This has the effect of varying the *effective wheel radius*, which is an important parameter in the kinematic models used for position estimation. Unfortunately, wheel slippage and sinkage are often difficult to measure and estimate in a straightforward manner.

Some progress has been made, however, in developing statistical estimation approaches for planetary rovers [27].

This case study addresses the problem of mitigating the effects of wheel slippage and sinkage through active management of traction via speed control, and effective wheel radius estimation. The target platform is the Pioneer-AT (All-Terrain), a commercially available mobile robot designed for rough terrain mobility. At JPL, it is utilized for research and development of perception and navigation concepts for eventual integration on more capable field robots. The factory configuration of the robot includes a low-profile chassis with a PID-controlled locomotion system of 4 wheels (driven by DC motors) and an array of ultrasonic range sensors, managed by an embedded 16 MHz MC68HC11-based microcontroller. The system was enhanced at JPL with additional onboard computing (Pentium II laptop) and a vision system for real-time terrain assessment (see Fig. 7). The laptop computer hosts all high-level intelligence and commands the low-level motor control system. The ultrasonic sensors are not utilized for this case study.

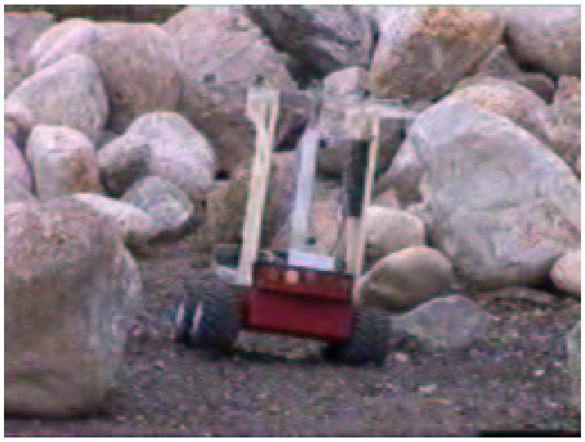


Fig. 7. Pioneer-AT rover with enhancements.

4.1 Motivation and Approach

Traction control solutions are often derived from analyses based on the following equation for wheel slip ratio, λ_s, which is defined non-dimensionally as a percentage of vehicle forward speed, v [28]:

$$\lambda_s = \left(1 - \frac{v}{r_w \omega_w}\right) \times 100. \tag{5}$$

Here, r_w is the nominal radius of the vehicle wheel and ω_w is the wheel rotational speed. Eq. 5 expresses the normalized difference between vehicle and wheel speeds. When this difference is non-zero, wheel slip occurs. The objective of traction control is to regulate λ_s to maximize traction. This is a relatively straightforward regulation task if v *and* ω_w are both observable. The wheel rotational speed ω_w is typically available from shaft encoders or tachometers. However, it is often difficult to measure the actual over-the-ground speed v for off-road wheeled vehicles. This is due to a lack of adequate practical sensing solutions as well as nonlinearities and time-varying uncertainties caused by wheel-soil interactions [29]. Effective solutions have been found for automotive applications, but in many of these cases, measurement of v is facilitated by an even surface on which the vehicle travels, or by special sensing arrangements engineered for the operating environment. In large part, the available automotive solutions are not directly transferable to off-road vehicle applications. The traction management problem for off-road robotic vehicles can be addressed using a soft computing approach that does not rely on accurate sensing of over-the-ground vehicle speed to compute wheel slip. Instead, the strategy relies on visual perception of terrain surface texture to infer appropriate tractive speed controls.

Distinct terrain surfaces reflect different textures in visual images. The ability to associate image textures to terrain surface properties such as traction, hardness, or bearing strength is directly useful for autonomous traction management. To provide this capability, we make use of an onboard monocular camera pointed such that its field-of-view (FOV) covers the ground area in front of the rover as illustrated in Fig. 8. On the Pioneer-AT shown in Fig. 7,

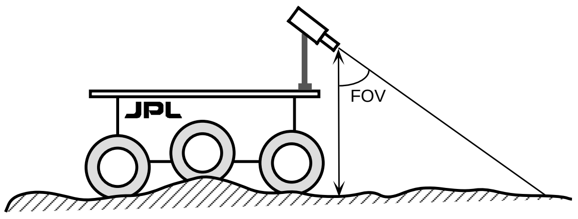


Fig. 8. Camera mounted on rover.

the ground-facing camera on the front of the rover is mounted $0.3\,m$ above the ground, tilted downward 45^o with a 45^o FOV. This camera enables surface traction classification using soft computing-based image analysis. (Cameras shown mounted on the raised platform are used for strategic navigation.) In particular, hard computing methods of computer vision are combined with an image texture classification approach using an artificial neural network (ANN). This provides an automated method of classifying the terrain surface just ahead of the rover with respect to tractive quality. The ANN is adopted to realize this capability due to its effectiveness for representing arbitrary input-output relationships. This qualitative knowledge serves as input to a supervisory fuzzy logic speed controller, which produces speed recommendations associated with the current perception. The fuzzy logic implementation also serves to absorb the effects of potential uncertainties in quantities representing the traction classification and inferred speeds. Simple fuzzy speed control rules are formulated based on off-road driving heuristics for facilitating maintenance of wheel traction, thereby mitigating excessive wheel slip. For a variety of potential surface types, the maximum speeds achieved before the onset of wheel slippage (tractive speeds) are determined from empirical traction tests performed with the actual rover. Given this information, commanded vehicle speed can be modulated during traversal based on visual classification of the terrain surface type in front of the rover. This is analogous to the perception-action process that takes place when a human driver notices an icy road surface ahead and decelerates to maintain traction. The perception of the surface type is also used to roughly estimate the effective wheel radius used for dead-reckoning on classified terrain surface types. Details of the implementation follow.

4.2 Vision-Based Strategy and Implementation

Considering the typical surfaces that the rover may encounter, three different texture prototypes are selected to train the ANN: sand, gravel, and compacted soil. Classification of the different textures is achieved using the following strategy:

- Extract a set of 40×40 image blocks from image data.
- Reduce image data dimensionality using orthogonal sub-space projection.
- Train the ANN on a set of texture prototypes projected on the eigenvector set.
- During run-time, feed projected texture images to the trained ANN.
- Extract texture prototype output from network & classify ground surface type.

Assuming the section of the ground image just ahead of the front wheels is free of obstacles, a set of 40×40 pixel image blocks is randomly selected from a camera image of size 320×280 pixels. The assumption of an obstacle-free area immediately ahead is plausible since a separate strategic navigation module handles the more forward-looking function of guiding the vehicle towards traversable terrain; navigation details are reported in [30]. To reduce the large data dimensionality inherent in typical computer vision-based applications, a filtering step is performed using a standard technique called Principal Component Analysis (PCA) [31]. PCA is a linear optimal method for reducing data dimensionality by identifying the axis about which the desired feature set varies the most. This orthogonal sub-space projection of the image subset permits effective extraction of features embedded in the surface image data set in real time. This technique reduces the dimensionality of the image set while preserving as much of the signal as possible. PCA computes a set of orthonormal eigenvectors (filters) of a data set that captures the greatest correlation between features. The filters associated with a given feature set are derived from the distribution of potential dynamic features embedded in the images. To characterize the distribution of these features, the covariance matrix, R, is found for image subsets containing the desired dynamic features. The eigenvector problem, $R\mathbf{w} = \lambda\mathbf{w}$, is solved to derive the set of filters, $\mathbf{w}$, used in the algorithm outlined above to maximize the greatest correlation between features. A total of 30 eigenvectors is used to reduce the 40×40 image block (1600 pixel values) to a pattern set of 30 values (Fig. 9) by projecting the image data onto the most significant filters. This reduced data set is then used to train the ANN (Fig. 10) to associate texture with several surface types.

The ANN uses 30 input nodes corresponding to the projected image block and 1 output node representing the surface type classification; its hidden layer has 20 processing elements. Initially, the ANN is trained using backpropagation by finding a set of weights that produce the desired surface type classification for given training data input. After training the network on typical image data representing different texture prototypes imaged in different illumination conditions, it is utilized to classify surface types during run-time. For the algorithm, the network output provides the qualitative information needed to make any necessary adjustments to wheel speed in order to maintain traction on the classified surface. Fig. 11 shows several images of real terrain data properly classified by the trained ANN; these images were not included in the data set used to train the ANN.

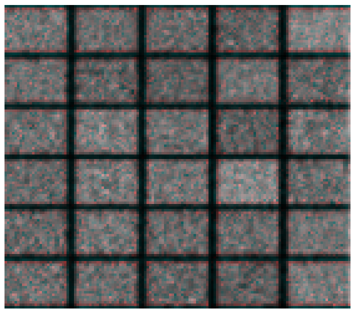


Fig. 9. Texture eigenvectors.

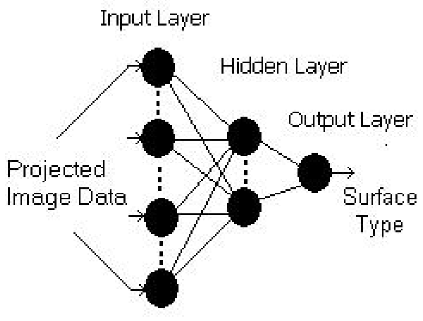


Fig. 10. ANN for surface classification.

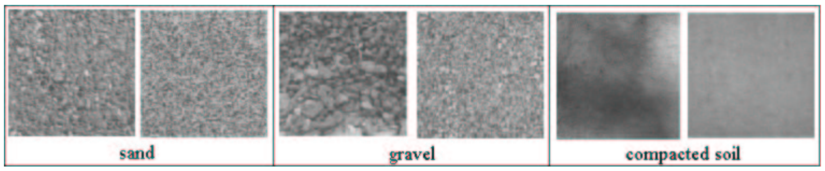


Fig. 11. Terrain surface images classified by the ANN.

Tractive Speed Modulation The ANN is trained to classify ground textures and produce surface type outputs represented as numerical values in the unit interval [0, 1], with 0 corresponding to surfaces of very low traction (e.g., ice) and 1 corresponding to surfaces of very high traction (e.g., dry cement). This is a design decision motivated by a desire to establish some *intuitive* correlation to actual wheel-terrain coefficients of friction. In this way, we can make a qualitative association between output of neural networks and expected terrain traction in front of the rover. We will refer to the texture prototype output as the *traction coefficient*, denoted by C_t. The range of traction coefficients, [0,1], obtained from the ANN is partitioned using three fuzzy sets with linguistic labels of LOW, $MEDIUM$, and $HIGH$ as shown in Fig. 12. Based on these definitions, a single-input-single-output fuzzy logic

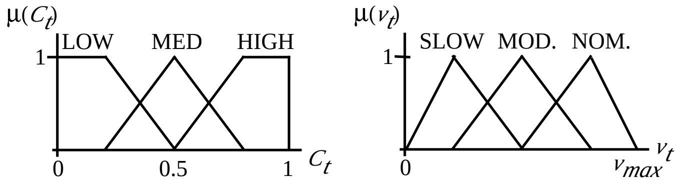


Fig. 12. Fuzzy sets for traction management.

controller infers tractive speed set-points, v_t, for input to the low-level PID wheel motor controller. The range of allowable speeds is partitioned using fuzzy sets labeled $SLOW$, $MODERATE$, and $NOMINAL$ as shown in Fig. 12, with v_{max} specified according to the application. The following fuzzy logic rules are applied to modulate tractive speeds in proportion to expected terrain traction in front of the rover:

- IF C_t is LOW, THEN v_t is SLOW.
- IF C_t is MEDIUM, THEN v_t is MODERATE.
- IF C_t is HIGH, THEN v_t is NOMINAL.

Complete definition of membership functions for tractive speeds v_t is based on results of prior traction tests as mentioned earlier. As such, these membership function definitions are vehicle-dependent and reflect knowledge derived about non-slip speeds achieved when the vehicle was tested on various terrain surfaces. For best results, traction tests should be performed on surfaces that represent the expected roughness, hardness, and slope variations of the rover operating environment. Once membership functions are defined, the robot is equipped to perceive and react to learned terrain conditions in a similar manner as a human driver. A more sophisticated approach to perception and reaction on very challenging terrain is proposed in [7], and is better suited for articulated rover mobility mechanisms that are capable of kinematic reconfiguration. That strategy uses look-ahead stereo visual perception as well as onboard kinematic analysis of potential tip-over and expected traction to plan subsequent stable and tractive configurations.

Effective Wheel Radius Modulation The sophistication of localization methods for outdoor mobile robots depends on the available sensor suite onboard the vehicle. A variety of viable hard computing solutions exist, many of which employ several sensor types (e.g., wheel odometry, inertial measurement units, GPS, visual odometry, sun sensing, etc.) and perform sensor fusion for position and attitude estimation, often based on Kalman filter formulations [32–35]. The vehicle employed in this case study uses odometry based on wheel encoder data as the primary means of position estimation. Such dead-reckoning is prone to accumulated error during traversal. As such, it is not recommended as a sole means of position estimation for field mobile robots over significant distances. In lieu of more sophisticated estimation methods due, for example, to limitations of onboard sensing, it is highly desirable to adopt techniques for improving dead-reckoning accuracy. One such technique is a wheel velocity synchronization approach described in [32], which is designed for improved odometry and power efficiency of 6-wheeled rovers with independently-driven wheels. As each wheel experiences different loading profiles, the wheel velocity synchronization serves to minimize motion interactions between them which may cause excessive wheel slippage and the tendency to side-slip as the suspension system traverses over obstacles. The Pioneer-AT platform used in this case study is kinematically simpler. The two wheels on each side of the platform are physically coupled for synchronized drive motion, and the platform is steered differentially due to relative action of the paired wheels on either side.

For the simpler Pioneer-AT, the traction coefficient described above can be used to address a more fundamental aspect of the position estimation problem, namely, position error reduction via intelligent estimation of effective wheel radius. As mentioned earlier, the accuracy of kinematic models used to compute rover localization updates depends on the specification of a nominal wheel radius, r_w. This kinematic parameter is used to compute the equivalent linear distance $r_w \theta_w$ traveled by a wheel after any rotational wheel displacement θ_w on the terrain. For the Pioneer-AT, these linear distances are used to compute robot position and heading updates in a reference coordinate frame according to the following kinematic equations:

$$x_{k+1} = x_k - r_w \left(\frac{\theta_{w,r} + \theta_{w,l}}{2} \right) \sin \phi_{k+1}, \tag{6a}$$

$$y_{k+1} = y_k + r_w \left(\frac{\theta_{w,r} + \theta_{w,l}}{2} \right) \cos \phi_{k+1}, \tag{6b}$$

$$\phi_{k+1} = \phi_k + r_w \left(\frac{\theta_{w,r} - \theta_{w,l}}{d} \right). \tag{6c}$$

Here, x and y are Cartesian coordinates of the robot position on the terrain surface and ϕ is its heading; $\theta_{w,r}$ and $\theta_{w,l}$ are the rotational displacements of the right and left wheels (measured by encoders in the two front wheels), and d is the lateral distance between them. The displacements of wheels on the same side are assumed to be equal due to the physical coupling of the Pioneer-AT locomotion mechanism. The influence of r_w on the localization update is apparent in these equations. For compliant wheels/tires such as those on the Pioneer-AT, r_w is reduced as the tire compresses in normal reaction with the terrain to an effective wheel radius, r_{eff}, which should be used instead of r_w in Eqs. (6a)-(6c) to yield more accurate updates. Non-compliant wheels produce a similar effect. For example, as a vehicle with non-compliant wheels traverses terrain with both hard-packed soil and soft sand, use of r_w in the kinematic model is valid only over the hard-packed terrain. As the terrain load-bearing strength decreases (over softer soil), so does the effective wheel radius, r_{eff}, and the accuracy of the model. To reduce the effect of propagating this non-systematic error during rover traverses on varied terrain, we make further use of C_t to estimate the varying r_{eff} according to the following linear regression relationship,

$$r_{eff} = r_w C_t + c_{reg}, \tag{7}$$

where c_{reg} is a regression constant, which can be evaluated for a given vehicle via regression analysis of empirical data produced by multiple runs over varied terrain surfaces. Here, linear regression is assumed to produce a sufficient estimation model. However, an effective wheel radius estimation model for a given vehicle and terrain could require nonlinear regression to achieve desired improvements. It should be noted that this may still be insufficient depending upon the localization accuracy required in a given application. With the luxury of additional onboard sensing, such as an inertial measurement unit (including rate gyroscope and accelerometers), one could improve the position estimate further by employing the well-known hard computing techniques of Kalman filtering [2]. A good example of this is provided in [33] where a Kalman filter is formulated to improve localization for the Pioneer-AT robot.

This case study describes a low-level vision-based strategy for terrain perception with supervisory-level fuzzy control of tractive vehicle speed and terrain-aided position estimation. Statistical techniques [36] and color-based methods [37] for terrain classification have also been developed at JPL for navigation and mobility in vegetated terrain. These applications employ laser rangefinders as well as color and infrared stereo cameras for day and night vision, and the target platforms are cross-country military unmanned ground vehicles. In the next section, Case Study 3 covers an application of fuzzy logic speed control for a military all-terrain autonomous vehicle that differs significantly in size and propulsion from the field robots discussed thus far. The soft computing strategy, employed at a lower level, serves to overcome the lack of a complete analytical model.

5 Case Study 3: Adaptive Fuzzy ATV Speed Control

In this case study, an adaptive fuzzy speed control design for a throttle-regulated internal combustion engine on an All Terrain Vehicle (ATV) is presented. The ATV, shown in Fig. 13 is one of several mobile platforms used in the *CyberScout* project at Carnegie Mellon University, which aims to develop distributed mobile robotics technologies that will extend the sphere of awareness and mobility of small military units. The low-level ATV speed controller is one of the subsystems crucial to achieving complete autonomy.

Various automatic speed controllers based on hard computing techniques alone (e.g., adaptive, robust, and sliding model) have been implemented on automobiles. However, the proposed control techniques are not directly applicable to the ATV throttle control problem for the following reasons. First, the ATV engine is mechanically controlled via a carburetor, unlike most automobiles, which have microprocessor-based engine management systems that guarantee maximum engine efficiency and horsepower. Second, the ATV carburetor clearance makes it difficult to incorporate a sensor to measure the throttle plate angle, which is required in virtually all of the automotive speed controllers reported in the literature. Third, and most importantly, automobile cruise control does not work well at speeds below 30 miles per hour (mi/h) due to engine nonlinear torque and speed fluctuations. Finally, the ATV throttle is actuated via an R/C servo, with no explicit position feedback, instead of a pneumatic actuator which is the preferred actuator in most automobiles. The adaptive fuzzy throttle control strategy is applied to address these constraints for the ATV. Fuzzy logic rules are formulated from extensive experiments conducted by human operators and based on quantitative data. An adaptive control law is applied to augment the soft computing based control.

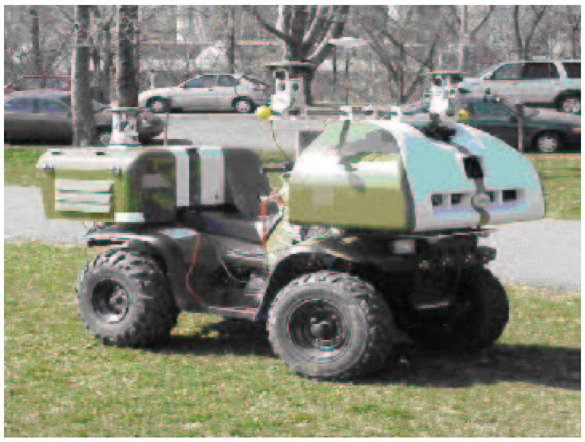


Fig. 13. Autonomous CyberATV "Lewis".

5.1 Motivation and Approach

The intended military operations task for the ATV requires that the speed control system be designed to provide smooth throttle movement, zero steady-state speed error, constant vehicle speed over varying road slopes, and robustness to system variations and operating conditions for a 2-30 mi/h speed range. The two main challenges in designing an effective speed controller for the ATV are: 1) the lack of a complete mathematical model for the engine, and 2) the highly nonlinear nature of the engine dynamics, especially for the targeted low speed range of 3-30 mi/h. Both of these factors make the use of classical control strategies such as PID control ineffective. To elaborate, initial open-loop experiments conducted on level terrain revealed that humans could not easily drive the ATV at speeds below 10 mi/h, shedding light on the nature of the nonlinear relationship between throttle valve openings and speed. Also, the throttle valve-opening threshold for initiating vehicle movement varied from one trial to the next, indicating a shifting operating point. Attempts to apply conventional PID control revealed that a PI controller could be used for higher speeds where the carburetor operation is fairly linear, i.e., speeds above 15 mi/h, thus covering the upper portion of the target speed range. (The measured speed signal is very noisy, so it was not feasible to implement a derivative component for a PID controller.) This result indicates that a possible approach is to use more than one control strategy via lookup tables, depending on the speed range. However, it is very difficult to apply conventional control techniques for the ATV since a complete mathematical model of the engine is not available, and developing this model requires information about the engine, which the manufacturer was unwilling to provide. An alternative approach is fuzzy logic control (FLC), since human experience combined with the extensive quantitative and qualitative results can be employed effectively in fuzzy systems. It was found that while fuzzy control provided very smooth throttle movement, it was insufficient for achieving the required steady-state accuracy, particularly for substantial changes in the terrain (up and down hill). Several attempts to tune the fuzzy membership functions did not significantly reduce the steady state error, suggesting the need for adaptivity. Since the FLC strategy worked fairly well, an adaptive control law based on that strategy was considered. Details of the fuzzy control strategy and adaptive control design are discussed below.

5.2 Fuzzy Speed Control Design and Implementation

The design goal for the ATV speed control is to minimize the magnitude of the speed error, ε, defined as the difference between desired and actual speed. Human operators can control the speed of the ATV via a throttle lever, which opens and closes the throttle valve to increase or reduce the speed of the ATV. Based on familiarity with the speed response to this action, fuzzy rules were formulated using speed error and change in control input to the throttle actuator. Automatic actuation of the ATV throttle is achieved via a Futaba R/C servomotor (added to the system) which is controlled by pulse width modulation (PWM) such that pulse widths of $1\,ms$, $1.5\,ms$, and $2\,ms$ correspond to idle, half, and full throttle, respectively. Speed feedback is obtained via a tachometer mounted in the gearbox. The change in throttle control is defined using the two past values of the control input, u, and can be expressed as $\Delta u = u_{(k-1)T} - u_{(k-2)T}$ where discrete time $t = kT$, $k = 0, 1, 2, \ldots, n$ and T is the sampling and control update period.

Fuzzy input and output membership functions are shown in Fig. 14. The five linguistic labels for speed error input (ε) are: Negative Large (NL), Negative Small (NS), Zero, Positive Small (PS), and Positive Large (PL). Three linguistic labels are similarly defined for the input change in throttle opening (Δu). The five linguistic labels for the output throttle opening (u) are: Zero, Small (SM), Medium (MED), Large (LG) and Very Large (VLG). The ranges of these linguistic variables were determined by experimentation and the physical constraints of the sensors employed, e.g., the R/C servomotor input pulse width command range of 1-2 ms. The Zero membership function center for throttle opening is defined to be slightly above idle engine speed. The FLC was implemented using product inference and a center-average defuzzifier [38]. Fig. 15a shows the complete set of fuzzy rules for u in tabular form along with a block diagram of the control structure (Fig. 15b), in which the derivative block represents Δu. Rules in the table of Fig. 15a can be interpreted in linguistic form. For example, the rule specifying "Very Large" for the throttle opening may be written as: $IF\,\varepsilon\,is\,PL\,and\,\Delta u\,is\,Zero, THEN\,u\,is\,VLG$. The fuzzy linguistic label names used here give an intuitive sense of how the rules apply. Experimentation and tuning of the membership functions, revealed that this rule set was sufficient to encompass all realistic combinations of inputs and outputs. Recall, however, that this fuzzy control was not sufficient to achieve the required steady-state accuracy alone. The theoretical formulation of the adaptive fuzzy control is outlined below.

5.3 Adaptive Fuzzy Control Solution

Assume that the rule base consists of multiple-input single-output rules of the form $R^{(j)}$: $IF\,x_1\,is\,A_1^j\,and \ldots and\,x_n\,is\,A_n^j, THEN\,y\,is\,C^j$, where $\mathbf{x} = (x_1 \ldots x_n) \in N$, $y_j \in S$ denotes the linguistic variables associated with inputs and outputs of the fuzzy system. A_i^j and C^j are linguistic values of linguistic variables $\mathbf{x}$ and y in the universes of discourse N and S respectively; $j = 1, 2, \ldots, Q_R$ (number of rules). A fuzzy system consisting of a singleton fuzzifier, product inference, center-average defuzzifier and triangular membership functions can be written as [38]

$$f(\mathbf{x}) = \frac{\sum_{j=1}^{Q_R} \bar{y}^j \left[\prod_{i=1}^{n} \mu_{A_i^j}(x_i)\right]}{\sum_{j=1}^{Q_R} \left[\prod_{i=1}^{n} \mu_{A_i^j}(x_i)\right]}, \tag{8}$$

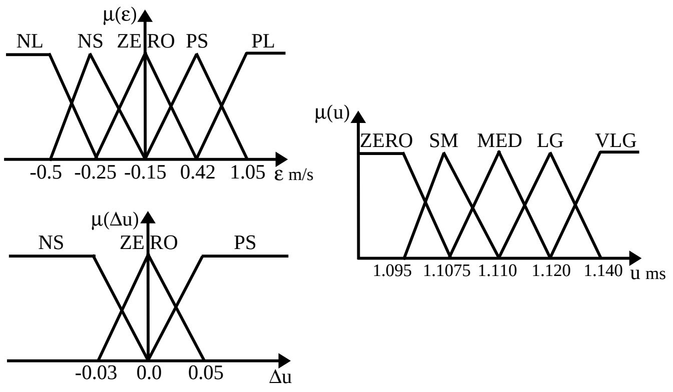
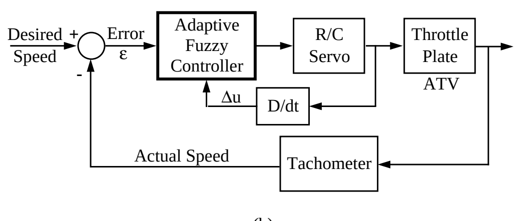


Fig. 14. ATV speed control input and output membership functions.

ε \ Δu	NS	ZERO	PS
NL	ZERO	ZERO	ZERO
NS	SMALL	SMALL	SMALL
ZERO	ZERO	ZERO	ZERO
PS	SMALL	LARGE	MEDIUM
PL	MEDIUM	VERY LARGE	LARGE

(a)



(b)

Fig. 15. (a) Fuzzy rule table for ATV speed control; (b) Controller block diagram.

where $f : N \subset \Re^n \to \Re$, $\mathbf{x} = (x_1 \dots x_n)^T \in N$, $\mu_{A_i^j}(x_i)$ is a triangular membership function and $\bar{y}^j$ is the point in S where μ_{C^j} is maximum or equal to 1. If $\mu_{A_i^j}(x_i)$ and $\bar{y}^j$ are free (adjustable) parameters, then (8) can be written as

$$f(\mathbf{x}) = \vartheta^T \mathbf{\Psi}(\mathbf{x}), \tag{9}$$

where $\vartheta = (\bar{y}^1 \dots \bar{y}^{Q_R})$ is a parameter vector and $\mathbf{\Psi}(\mathbf{x}) = (\psi^1(\mathbf{x}) \dots \psi^{Q_R}(\mathbf{x}))^T$ is a regression vector with the regressor given by

$$\psi_i(\mathbf{x}) = \frac{\prod_i^n \mu_{A_i^j}(x_i)}{\sum_{j=1}^{Q_R}(\prod \mu_{A_i^j}(x_i))}. \tag{10}$$

Eq. (9) is referred to as an adaptive fuzzy system [38]. There are two main reasons for using adaptive fuzzy systems as building blocks for adaptive fuzzy controllers. Firstly, they have been proven to be universal function approximators [38]. Secondly, all the parameters in $\mathbf{\Psi}(\mathbf{x})$ can be fixed at the beginning of the adaptive fuzzy system expansion design procedure, so that the only free design parameters are ϑ. In this case $f(\mathbf{x})$ is linear in the parameters.

The advantage of this approach is that very simple linear parameter estimation methods can be used to analyze and synthesize the performance and robustness of adaptive fuzzy systems. If no linguistic rules are available, the adaptive fuzzy system reduces to a standard nonlinear adaptive controller. This approach is adopted to synthesize the adaptive control law.

Adaptive Law Synthesis and Implementation From theoretical considerations and quantitative data, the following model was developed for the ATV engine [40]:

$$\begin{bmatrix} \dot{N} \\ \ddot{N} \end{bmatrix} = \begin{bmatrix} \frac{-d_1}{J_e} & 0 \\ \frac{-c_t c_e^2 \eta_{vol}^2 m_a}{J_e} & \frac{-d_1}{J_e} \end{bmatrix} \begin{bmatrix} N \\ \dot{N} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{-c_t c_e \eta_{vol} D_1}{J_e} \end{bmatrix} \alpha + \mathbf{E} \qquad (11)$$

where N is the engine speed (RPM); d_1 and D_1 are respectively engine friction and airflow constants; J_e is the engine moment of inertia; c_e is derived from the intake manifold and engine displacement; c_t is derived from the engine efficiency, combustion heat, and air-to-fuel ratio; η_{vol} is the engine volume efficiency; m_a is the net manifold air-mass flow rate; α is the throttle plate angle; and E lumps together higher-order cross-coupling terms and load torque. This engine model (11) represents a primitive mathematical model of internal combustion engine dynamics and by no means captures all the nonlinear parameters of the engine. The model can be expressed as

$$\dot{\mathbf{z}} = \mathbf{A}\mathbf{z} + \mathbf{B}\mathbf{u} + \mathbf{E}(\mathbf{z}) \qquad (12)$$

where $\mathbf{A}$ is Hurwitz and $\mathbf{E}(\mathbf{z})$ is the uncertainty in the model expressed as a function of the state $\mathbf{z}$. Therefore, there exists a unique positive definite matrix $\mathbf{P}$ that satisfies the Lyapunov equation

$$\mathbf{A}^T\mathbf{P} + \mathbf{P}\mathbf{A} = -\mathbf{Q} \qquad (13)$$

If the control input, $\mathbf{u}$, is expressed as an adaptive fuzzy system then (12) becomes,

$$\dot{\mathbf{z}} = \mathbf{A}\mathbf{z} + \mathbf{B}\vartheta^T\psi(\mathbf{z}) + \mathbf{E}(\mathbf{z}) \qquad (14)$$

Let [39],

$$\dot{\hat{\mathbf{z}}} = \mathbf{A}\hat{\mathbf{z}} + \mathbf{B}\vartheta^{*T}\psi(\hat{\mathbf{z}}) \qquad (15)$$

be the ideal engine model with no uncertainty with $\varepsilon = \mathbf{z} - \hat{\mathbf{z}}$, where ϑ^* denotes the optimal parameter vector. Therefore,

$$\dot{\varepsilon} = \mathbf{A}\varepsilon + \mathbf{B}\phi^T\psi(\varepsilon) + \hat{\mathbf{E}} \qquad (16)$$

where $\phi = \vartheta - \vartheta^*$, and $\hat{\mathbf{E}}$ is an estimate of the upper bounds of the uncertainties. The following Lyapunov function (with $\gamma > 0$ as a design parameter) is used to derive the adaptive control law given by Eq. (18) below, which ensures that $\varepsilon \to \mathbf{0}$ as $t \to \infty$ (see [40] for details on this formulation).

$$V = \frac{1}{2}\left(\varepsilon^T \mathbf{P} \varepsilon + \frac{\phi^T \phi}{\gamma ||\hat{\mathbf{E}}||}\right) \tag{17}$$

$$\dot{\phi} = -\gamma ||\hat{\mathbf{E}}|| \, ||\varepsilon^T \mathbf{P} \mathbf{B}|| \, \psi(\varepsilon) \tag{18}$$

The fuzzy rule table of Fig. 15a was used to implement (18) for adaptive fuzzy speed control on the ATV. The insight gained from the non-adaptive FLC was used to select the ϑ values to lie within the interval [1, 2]. The remaining control parameters were set as follows: $\mathbf{Q} = diag(3,3)$, $\hat{\mathbf{E}} = [120 \;\; 0]^T$, $\gamma = 0.00025$; and $\mathbf{\Psi}(\varepsilon)$ was formulated using the IF part of fuzzy rules in Fig. 15a. Finally, $\mathbf{P}$ is obtained from the iterative solution of (13), and $\hat{\mathbf{E}}$ and γ are obtained empirically.

Typical ATV performance with the adaptive fuzzy control law is shown in Figs. 16 and 17, which depict representative ATV responses to selected speed commands ($2.97\,mi/h$ and $3.4\,mi/h$). Acceptable steady-state error performance is achieved; in addition, considerable improvement over the non-adaptive FLC was observed with respect to disturbance rejection (load and terrain). The adaptive algorithm responds to varying terrain by continuously minimizing the speed error by tuning the center of the input membership functions.

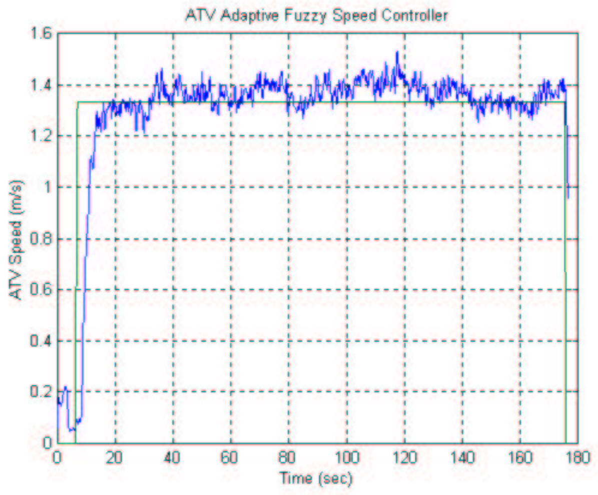


Fig. 16. ATV speed response to $1.3\,m/s$ command.

At very slow speeds for the ATV (e.g., $1.0\,m/s$), the speed response overshoot and settling times tend to increase. Since the ATV is back-heavy, considerable momentum is required to initiate motion on slight inclines, and brief

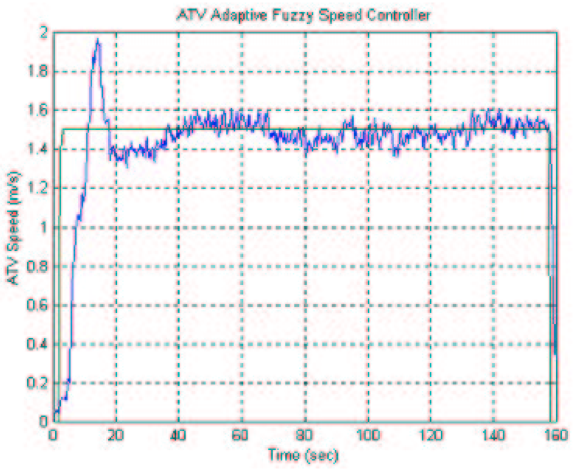


Fig. 17. ATV speed response to 1.5 m/s command.

slippage in its automatic transmission causes considerable drops in speed to well below the commanded speed. In such cases, the adaptive fuzzy control responds with small increases in throttle input (rather than the large response that a PI controller would typically generate) and regains the commanded velocity without significant overshoot once the slipping ceases. For this case study, the combined soft and hard computing-based control solution successfully achieves the desired properties of smooth and accurate control at speeds within the desired range, as well as robustness to varying terrain.

6 Discussion and Conclusion

Autonomous robotic operations in natural outdoor environments require robots to perform complex tasks for which analytical models of sufficient accuracy or completeness are not always available. To address such issues while producing reliable and safe field robotic systems it is often advantageous to supplement conventional hard computing methods with particular soft computing techniques. This chapter presented several case studies in this regard describing soft computing techniques that have been integrated on physical field mobile robot systems and validated in natural terrain environments.

Case study 1 focused on a high-level vision-based strategy for terrain perception using efficient stereo image processing to facilitate rover navigation. A hybrid wavelet processing and neural network system was described which generates appropriate navigation actions based on input of a stereo pair of images. The system reacts to a pattern of wavelet coefficients in much the same way as binocular neurons in the mammalian visual pathways. Case Study 2 described a terrain perception strategy that employs low-level monocular vision to facilitate maintaining mobility performance during navigation using supervisory-level control of tractive vehicle speed. This case also presented a fundamental approach to improving position estimation enabled by the same perception system. The techniques applied in Case Study 2 may be preferred for systems with minimal onboard sensing and computing resources. Case Study 3 presented an adaptive fuzzy throttle control design for controlling speed of an autonomous ATV driven by an internal combustion engine. The control algorithm has been implemented on two other ATVs with very little recalibration of the control parameters. The formulation of the fuzzy rules for the system may be relevant to other practical applications where a complete mathematical model is not available.

Soft computing has been successfully applied to address perception using neural networks based on both fuzzy self-organizing feature maps and backpropagation, as well as supervisory and low-level speed control using rule-based fuzzy logic and adaptive fuzzy systems, respectively. However, implementation experiences associated with the case studies have revealed some disadvantages of applying these soft computing methods for perception and control. A key disadvantage, from the point of view of the perception applications, relates to specification of the initial knowledge that is embedded within the system and ultimately used as the basis for motion decisions. For example, the generalization capability (and therefore, success) of the vision-based neural networks depends on the richness of a set of training images and how well they represent the target environments. If the training set is insufficient, there may be visual scenes for which the system has not learned appropriate motion responses. It is not always obvious when this is the case, and extensive testing may not reveal such insufficiencies. The strategies in this chapter make use of empirical tests and adaptive learning techniques in order to ensure that once initial knowledge is established, it can be improved based on actual system behavior in subsequent tests. By allowing adaptive capability, the initial establishment of a thorough knowledge base is not as critical if significant testing is performed.

Regarding control applications, soft computing techniques have been subject to debate and skepticism, since they do not have the formal rigor of conventional control techniques. This can be attributed to the fact that most successful applications of soft computing are based on simulation and experimental results, rather than the theoretical proofs relied upon by classical control theorists and practitioners. Additionally, most soft computing applications have been in mass non-critical consumer goods with less stringent performance criteria than the traditional control areas of aerospace and industrial systems. Nevertheless, soft computing has been demonstrated to solve some practical problems that have been beyond the reach of conventional control techniques. The rapid growth of research in this area establishes the fact that there is a genuine need for soft computing-based control strategies. While technological tools exist for implementing them, tools for synthesis of performance criteria, robustness, and design methodology are lacking, however. Hence, there is a need for further research in this direction. Until techniques and tools can be developed that can subject soft computing control systems to such factors, the debate and skepticism are likely to remain.

The computing strategies described herein for perception and control were developed for application on wheeled robots to be potentially deployed for space and military missions. The case studies serve only as existence proofs of the concepts presented, as well as practical implementation examples validated on research prototype vehicles. For real missions, various requirements and operability or safety restrictions are specified on a task by task basis. Compliance of combined hard and soft computing strategies with real mission constraints must be verified and validated through extensive testing prior to field deployments. Note that the utility of the strategies presented is not limited to the specific field robots described in this chapter. Indeed, the soft computing strategies may be adapted and integrated with hard computing solutions on a wider variety of autonomous field robots to address common problems of perception, navigation, and control. In practice, it is typically the *solutions* to the common problems that must be tailored for specific robotic systems. Particular design and implementation decisions must be governed by the real-world task constraints, mission objectives and requirements, and/or vehicle capabilities.

Acknowledgment

The rover research described in this chapter was performed at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration. The ATV research described in this chapter was performed at Carnegie Mellon University, sponsored by DARPA contract F04701-97-C-0022, "An Autonomous, Distributed Tactical Surveillance System." Publication support was provided by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with NASA.

References

1. Fu, K. S., Gonzalez, R. C., Lee, C. S. G. (1987): Robotics: Control, Sensing, Vision, and Intelligence. McGraw-Hill, New York
2. Dudek, G., Jenkin, M. (2000): Computational Principles of Mobile Robotics. Cambridge University Press, Cambridge, UK
3. Driankov, D., Saffiotti, A., (Eds.) (2001): Fuzzy Logic Techniques for Autonomous Vehicle Navigation. Fuzziness and Soft Computing Series **61**, Physica-Verlag, Heidelberg
4. Thrun, S. (2000): Probabilistic Algorithms in Robotics. Tech. Report CMU-CS-00-126, Computer Science Department, Carnegie Mellon University, Pittsburgh, PA
5. Dubowsky, S., Farritor, S., et al. (1996): Research on Field and Space Robotic Systems. In: Proc. ASME Computers in Engineering Conf., Irvine, CA

6. Hagras, H., Callaghan, V., Colley, M. (2001): Outdoor Mobile Robot Learning and Adaptation. IEEE Robotics & Automation Magazine **8**, 3, 53–69
7. Schenker, P. S., Pirjanian, P., et al. (2000): Reconfigurable Robots for All Terrain Exploration. In: McKee, G. T., Schenker, P. S. (Eds.): Proc. SPIE **4196**, Sensor Fusion and Decentralized Control in Robotic Systems III. Boston, MA
8. Gauss, V. A., Bay, J. S. (1998): A Fuzzy Logic Solution for Navigation of an Autonomous Subsurface Planetary Exploration Robot. In: Proc. IEEE Intl. Symp. on Intelligent Control, Gaithersburg, MD, 559–564
9. Kaplan, M. H. (1976): Modern Spacecraft Dynamics and Control. John Wiley & Sons, New York
10. Mishkin, A., Morrison, J., Nguyen, T., et al. (1998): Experiences with Operations and Autonomy of the Mars Pathfinder Microrover. In: Proc. IEEE Aerospace Conf., Aspen, CO
11. Matthies, L. (1992): Stereo Vision for Planetary Rovers: Stochastic modeling to near real-time implementation. Intl. J. Computer Vision **8**, 71–91
12. Belhumeur, P. N. (1993): A Binocular Stereo Algorithm for Reconstructing Sloping, Creased, and Broken Surfaces in the Presence of Half-occlusion. In: Proc. Intl. Conf. on Computer Vision. New York, NY, 431–438
13. Churchland, P. S., Sejnowski, T. J. (1993): The Computational Brain. MIT Press, Cambridge, MA
14. Daugman, J. G. (1985): Uncertainty Relation for Resolution in Space, Spatial Frequency, and Orientation Optimized by Two-dimensional Visual Cortical Filters. J. Optical Society of America A **2**, 1160–1169
15. Hubel, D. H., Wiesel, T. N. (1963): Shape and Arrangement of Columns in the Cat's Striate Cortex. J. of Physiology **165**, 559–568
16. Dobbins, A., Zucker, S. S., Cynader, M. S. (1987): Endstrapped Neurons in the Visual Cortex as a Substrate for Calculating Curvature. Nature **329**, 438–441
17. Mallat, S., Zhong, S. (1992): Characterization of Signals from Multiscale Edges. IEEE Trans. PAMI **14**, 710–732
18. Andersson, L., Hall N., Jawerth, B. et al (1994): Wavelets on Closed Subsets of the Real Line. In: Schumacher, L. L., Webb, G. (Eds.): Topics in the Theory and Applications of Wavelets. Academic Press, New York, NY
19. Cohen, A., Daubechies, I., Jawerth, B., Vial, P. (1993): Multiresolution Analysis, Wavelets and Fast Algorithms on an Interval. Comptes Rendes **316, I**, 417–421
20. Hoffman, B. D., Baumgartner, E. T., Huntsberger, T., Schenker, P. S. (1999): Improved Rover State Estimation in Challenging Terrain. Autonomous Robots **6**, 2, 113–130
21. Huntsberger, T. L. (1990): Comparison of Techniques for Disparate Sensor Fusion. In: Proc. SPIE Symp. Sensor Fusion III: 3-D Perception and Recognition **1383**, Boston, MA, 589–595
22. Kohonen, T. (1989): Self-Organization and Associative Memory, 3rd Edition. Springer-Verlag, Berlin
23. Pal, N. R., Bezdek, J. C., Tsao, E. C.-K. (1993): Generalized Clustering Networks and Kohonen's Self-organizing Scheme. IEEE Trans. Neural Networks **4**, 549–557
24. Bezdek, J. C. (1981): Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum Press, New York
25. Huntsberger, T. L., Jacobs, C. L., Cannon, R. L. (1985): Iterative Fuzzy Image Segmentation. Pattern Recognition **18**, 131–138

26. Huntsberger, T. L., Ajjimarangsee, P. (1990): Parallel Self-organizing Feature Maps for Unsupervised Pattern Recognition. Intl. J. of General Systems **16**, 357–372
27. Wilcox, B. H. (1994): Non-geometric Hazard Detection for a Mars Microrover. In: NASA/AIAA Conference on Intelligent Robotics in Field, Factory, Service, and Space. Houston, TX, 675–684
28. Gillespie, T. D. (1992): Fundamentals of Vehicle Dynamics. Society of Automotive Engineers, Inc.
29. Tunstel, E., Howard, A. (2002): Approximate Reasoning for Safety and Survivability of Planetary Rovers. Fuzzy Sets and Syst. Elsevier Science, In press.
30. Howard, A., Seraji, H. Tunstel, E. (2001): A Rule-based Fuzzy Traversability Index for Mobile Robot Navigation. In: IEEE Intl. Conf. Robotics and Automation. Seoul, Korea, 3067–3071
31. Diamantaras, K., Kung, S. Y. (1993): Principal Component Neural Networks: Theory and applications. John Wiley & Sons, New York
32. Baumgartner, E. T., Aghazarian, H., Trebi-Ollennu, A. (2001): Rover Localization Results for the FIDO Rover. In: Proc. SPIE Conf. Sensor Fusion and Decentralized Control in Autonomous Robotic Systems IV **4571**. Newton, MA
33. Goel P., Roumeliotis S. I., Sukhatme G. S. (1999): Robust Localization using Relative and Absolute Position Estimates. In: Proc. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems. Kyongju, Korea, 1134–1140
34. Barshan, B., Durrant-Whyte, H. F. (1995): Inertial Navigation Systems for Mobile Robots. IEEE Trans. Robotics and Automation **11**, 328–342
35. Olson, C. F., Matthies, L. H. et al (2001): Stereo Ego-motion Improvements for Robust Rover Navigation. In: Proc. IEEE Intl. Conf. on Robotics and Automation. Seoul, Korea, 1099–1104
36. Bellutta, P., Manduchi, R., Matthies, L., Rankin, A. (2000): Terrain Perception for DEMO III. In: Proc. Intelligent Vehicles Conf. Dearborn, MI
37. Macedo, J., Manduchi, R., Matthies, L. (2000): Ladar-based Discrimination of Grass from Obstacles for Autonomous Navigation. In: 7th Intl. Symp. on Experimental Robotics. Waikiki Beach, Hawaii
38. Wang, L-X. (1994): Adaptive Fuzzy Systems and Control Design and Stability Analysis. Prentice Hall, Englewood Cliffs, NJ
39. Trebi-Ollennu, A. (1996): Robust Nonlinear Control Designs using Adaptive Fuzzy Systems. Ph.D. thesis, Dept. Aerospace and Guidance Systems, School of Engrg, and Applied Science, Royal Military College of Science, Shrivenham, Cranfield University
40. Trebi-Ollennu, A., Dolan, J. M., Khosla, P. K. (2001): Adaptive Fuzzy Throttle Control for an All-terrain Vehicle. In: Proc. Instn. Mech. Engrs., Part I: J Syst. and Control Engrg. **215**, 189–198